AMENDMENTS TO THE CLAIMS

Please amend claims 1, 8, 9, 11, 14, 15, and 26-29, and insert new claims 30-41, as follow:

1. (Currently Amended) A computer-implemented method for processing a program statement in a database query language, the program statement corresponding to a plurality of operators, wherein an operator tree can be identified based upon the plurality of operators, the operator tree comprising a parent operator node~~, the parent operator node possibly associated with one or more child operator nodes~~, the method comprising:

~~(a)~~ identifying ~~whether one or more child nodes~~ a child node that is associated with the parent operator node ~~exist~~;

~~(b) for each of the identified one or more child nodes,~~ determining if the child node relates to an operator for which top-down processing can be performed;

~~(c)~~ calling and executing the ~~operators from (a)~~ operator for the child node ~~nodes that are eligible for top-down processing~~ to generate a result; and

~~(d) generating output results for a child node that is not eligible for top-down processing; and~~

~~(e)~~ outputting the ~~output results~~ result to a data stream without buffering the result.

2. (Original) The method of claim 1 further comprising:

determining whether the data stream already exists; and

creating the data stream if it does not exist.

3. (Original) The method of claim 1 in which the program statement is intended to create XML, wherein one or more XML tags are generated.

4. (Original) The method of claim 3 in which the program statement comprises a SQL/XML operator.

5. (Original) The method of claim 4 in which the SQL/XML operator is a XMLElement(), XMLAgg(), XMLConcat(), XMLForest(), XMLAttribute(), XMLComment(), or XMLPI() operator.

6. (Original) The method of claim 1 in which nodes corresponding to a concatenate operation or a CASE WHEN statement on top of SQL/XML operator are eligible for top-down processing.

7. (Original) The method of claim 1 in which the data stream is closed after the parent operator node has been fully evaluated.

8. (Original) The method of claim 1 ~~in which a~~, further comprising identifying another child operator node, wherein the another child operator node ~~is identified which~~ is not eligible for top-down processing.

9. (Currently Amended) The method of claim 8 in which the another child operator node ~~not eligible for top-down processing~~ is evaluated using bottom-up processing.

10. (Original) The method of claim 8 in which both top-down and bottom-up processing are used to evaluate the program statement.

11. (Currently Amended) The method of claim 1 in which the data stream is built at an intended target location ~~for the output results~~.

12. (Original) The method of claim 1 in which the data stream is a single data stream.

13. (Original) The method of claim 1 in which the data stream is built on a buffer, LOB, HTTP stream, segmented array, data socket, pipe, file, internet stream type, network stream type, or FTP stream.

14. (Currently Amended) The method of claim 1 in which an intermediate copy is not stored for the output ~~results~~ result.

15. (Currently Amended) A computer-implemented method for processing a program statement, the program statement corresponding to a plurality of operators, wherein an operator tree can be identified based upon the plurality of operators, the operator tree comprising a parent operator node, the method comprising:

(a) determining whether the parent operator node is related to a first child operator node that is eligible for top-down processing; and

(b) evaluating the first child operator node with top-down processing if the child operator is eligible for top-down processing, wherein the output from the first child operator node is output to a data stream without being buffered.

16. (Original) The method of claim 15 in which the program statement is intended to create XML, wherein one or more XML tags are generated.

17. (Original) The method of claim 16 in which the program statement comprises a SQL/XML operator.

18. (Original) The method of claim 17 in which the SQL/XML operator is a XMLElement(), XMLAgg(), XMLConcat(), XMLForest(), XMLAttribute(), XMLComment(), or XMLPI() operator.

19. (Original) The method of claim 15 in which nodes corresponding to a concatenate operation or a CASE WHEN statement over a SQL/XML operator are eligible for top-down processing.

20. (Original) The method of claim 15 in which an intermediate copy is not stored for the output from the first child operator node.

21. (Original) The method of claim 15 in which a second child operator node is identified which is not eligible for top-down processing.

22. (Previously Presented) The method of claim 21 in which the second child operator node not eligible for top-down processing is evaluated using bottom-up processing.

23. (Original) The method of claim 15 in which the data stream is built at an intended target location for the output from the first child operator node.

24. (Original) The method of claim 15 in which the data stream is a single data stream.

25. (Original) The method of claim 15 in which the data stream is built on a buffer, LOB, HTTP stream, segmented array, data socket, pipe, file, internet stream type, network stream type, or FTP stream.

26. (Currently Amended) A computer program product comprising a computer usable medium having executable code to execute a process for processing a program statement in a database query language, the computer usable medium comprising a volatile or non-volatile medium, the program statement corresponding to a plurality of operators, wherein an operator tree can be identified based upon the plurality of operators, the operator tree comprising a parent operator node~~, the parent operator node possibly associated with one or more child operator nodes~~, the process comprising:

~~(a)~~ identifying ~~whether one or more child nodes~~ a child node that is associated with the parent operator node ~~exist~~;

~~(b) for each of the identified one or more child nodes,~~ determining if the child node relates to an operator for which top-down processing can be performed;

~~(c)~~ calling and executing the ~~operators from (a)~~ operator for the child ~~nodes that are eligible for top-down processing~~ node to generate a result; and

~~(d) generating output results for a child node that is not eligible for top-down processing; and~~

~~(e)~~ outputting the ~~output results~~ result to a data stream without buffering the result.

27. (Original) A system for processing a program statement in a database query language, the program statement corresponding to a plurality of operators, wherein an operator tree can be identified based upon the plurality of operators, the operator tree comprising a parent operator node, ~~the parent operator node possibly associated with one or more child operator nodes~~, the method comprising:

~~(a)~~ means for identifying ~~whether one or more child nodes exist~~ a child node that is associated with the parent node;

~~(b)~~ means for determining if the child node relates to an operator for which top-down processing can be performed ~~for each of the identified one or more child nodes~~;

~~(c)~~ means for calling and executing the ~~operators from (a)~~ operator for the child ~~nodes that are eligible for top-down processing~~ to generate a result; and

~~(d) means for generating output results for a child node that is not eligible for top-down processing; and~~

~~(e)~~ means for outputting the output ~~results~~ result to a data stream without buffering the result.

28. (Currently Amended) A computer program product comprising a computer usable medium having executable code to execute a process for processing a program statement, the computer usable medium comprising a volatile or non-volatile medium, the program statement corresponding to a plurality of operators, wherein an operator tree can be identified based upon the plurality of operators, the operator tree comprising a parent operator node, the process comprising:

(a) determining whether the parent operator node is related to a first child operator node that is eligible for top-down processing; and

(b) evaluating the first child operator node with top-down processing if the child operator is eligible for top-down processing, wherein the output from the first child operator node is output to a data stream without being buffered.

29. (Original) A system for processing a program statement, the program statement corresponding to a plurality of operators, wherein an operator tree can be identified based upon the plurality of operators, the operator tree comprising a parent operator node, the method comprising:

(a) means for determining whether the parent operator node is related to a first child operator node that is eligible for top-down processing; and

(b) means for evaluating the first child operator node with top-down processing if the child operator is eligible for top-down processing, wherein the output from the first child operator node is output to a data stream without being buffered.

30. (New) The computer program product of claim 26, wherein the program statement is intended to create XML, wherein one or more XML tags are generated.

31. (New) The computer program product of claim 30, wherein the program statement comprises a SQL/XML operator.

32. (New) The computer program product of claim 31, wherein the SQL/XML operator is a XMLElement(), XMLAgg(), XMLConcat(), XMLForest(), XMLAttribute(), XMLComment(), or XMLPI() operator.

33. (New) The system of claim 27, wherein the program statement is intended to create XML, wherein one or more XML tags are generated.

34. (New) The system of claim 33, wherein the program statement comprises a SQL/XML operator.

35. (New) The system of claim 34, wherein the SQL/XML operator is a XMLElement(), XMLAgg(), XMLConcat(), XMLForest(), XMLAttribute(), XMLComment(), or XMLPI() operator.

36. (New) The computer program product of claim 28, wherein the program statement is intended to create XML, wherein one or more XML tags are generated.

37. (New) The computer program product of claim 36, wherein the program statement comprises a SQL/XML operator.

38. (New) The computer program product of claim 37, wherein the SQL/XML operator is a XMLElement(), XMLAgg(), XMLConcat(), XMLForest(), XMLAttribute(), XMLComment(), or XMLPI() operator.

39. (New) The system of claim 29, wherein the program statement is intended to create XML, wherein one or more XML tags are generated.

40. (New) The system of claim 39, wherein the program statement comprises a SQL/XML operator.

4'. (New) The system of claim 40, wherein the SQL/XML operator is a XMLElement(), XMLAgg(), XMLConcat(), XMLForest(), XMLAttribute(), XMLComment(), or XMLPI() operator.